K 6

```
FFFFFFFFFFFFFFF     OOOOOOOOO     RRRRRRRRRRRR     RRRRRRRRRRRR      TTTTTTTTTTTTTTT  LLL
FFFFFFFFFFFFFFF     OOOOOOOOO     RRRRRRRRRRRR     RRRRRRRRRRRR      TTTTTTTTTTTTTTT  LLL
FFFFFFFFFFFFFFF     OOOOOOOOO     RRRRRRRRRRRR     RRRRRRRRRRRR      TTTTTTTTTTTTTTT  LLL
FFF              OOO         OOO  RRR         RRR  RRR         RRR         TTT        LLL
FFF              OOO         OOO  RRR         RRR  RRR         RRR         TTT        LLL
FFF              OOO         OOO  RRR         RRR  RRR         RRR         TTT        LLL
FFF              OOO         OOO  RRR         RRR  RRR         RRR         TTT        LLL
FFF              OOO         OOO  RRR         RRR  RRR         RRR         TTT        LLL
FFF              OOO         OOO  RRR         RRR  RRR         RRR         TTT        LLL
FFFFFFFFFFFF     OOO         OOO  RRRRRRRRRRRR     RRRRRRRRRRRR            TTT        LLL
FFFFFFFFFFFF     OOO         OOO  RRRRRRRRRRRR     RRRRRRRRRRRR            TTT        LLL
FFFFFFFFFFFF     OOO         OOO  RRRRRRRRRRRR     RRRRRRRRRRRR            TTT        LLL
FFF              OOO         OOO  RRR   RRR        RRR   RRR               TTT        LLL
FFF              OOO         OOO  RRR   RRR        RRR   RRR               TTT        LLL
FFF              OOO         OOO  RRR   RRR        RRR   RRR               TTT        LLL
FFF              OOO         OOO  RRR      RRR     RRR      RRR            TTT        LLL
FFF              OOO         OOO  RRR      RRR     RRR      RRR            TTT        LLL
FFF              OOO         OOO  RRR      RRR     RRR      RRR            TTT        LLL
FFF                 OOOOOOOOO     RRR         RRR  RRR         RRR         TTT        LLLLLLLLLLLLLLL
FFF                 OOOOOOOOO     RRR         RRR  RRR         RRR         TTT        LLLLLLLLLLLLLLL
FFF                 OOOOOOOOO     RRR         RRR  RRR         RRR         TTT        LLLLLLLLLLLLLLL
```

```
**FILE**ID**FORERROR                                    K  4
```

```
FFFFFFFFFF    OOOOOO    RRRRRRRR    EEEEEEEEEE  RRRRRRRR    RRRRRRRR      OOOOOO    RRRRRRRR
FFFFFFFFFF    OOOOOO    RRRRRRRR    EEEEEEEEEE  RRRRRRRR    RRRRRRRR      OOOOOO    RRRRRRRR
FF          OO      OO  RR      RR  EE          RR      RR  RR      RR  OO      OO  RR      RR
FF          OO      OO  RR      RR  EE          RR      RR  RR      RR  OO      OO  RR      RR
FF          OO      OO  RR      RR  EE          RR      RR  RR      RR  OO      OO  RR      RR
FF          OO      OO  RR      RR  EE          RR      RR  RR      RR  OO      OO  RR      RR
FFFFFFFF    OO      OO  RRRRRRRR    EEEEEEEE    RRRRRRRR    RRRRRRRR    OO      OO  RRRRRRRR
FFFFFFFF    OO      OO  RRRRRRRR    EEEEEEEE    RRRRRRRR    RRRRRRRR    OO      OO  RRRRRRRR
FF          OO      OO  RR  RR      EE          RR  RR      RR  RR      OO      OO  RR  RR
FF          OO      OO  RR  RR      EE          RR  RR      RR  RR      OO      OO  RR  RR
FF          OO      OO  RR    RR    EE          RR    RR    RR    RR    OO      OO  RR    RR    ....
FF          OO      OO  RR    RR    EE          RR    RR    RR    RR    OO      OO  RR    RR    ....
FF            OOOOOO    RR      RR  EEEEEEEEEE  RR      RR  RR      RR    OOOOOO    RR      RR  ....
FF            OOOOOO    RR      RR  EEEEEEEEEE  RR      RR  RR      RR    OOOOOO    RR      RR  ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
FOR$
1-02

; Ro

; 4
```

```
0 MODULE FOR$$ERROR (%TITLE 'Internal FORTRAN error handling module'
0                 IDENT = '1-022'           ! File: FORERROR.B32  Edit: SBL1022
0                 ) =
1 BEGIN
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1
1 !++
1 ! FACILITY:  FORTRAN support library
1 !
1 ! ABSTRACT:
1 !
1 !       This module contains the error handlers needed by
1 !       the common OTS for handling FORTRAN errors.  In particular
1 !       there is a handler for errors in OPEN/CLOSE where ERR=
1 !       means error return to caller rather than a transfer.
1 !       A second handler (FOR$$ERR_END_HND is provided
1 !       for I/O statements where the optional ERR= and END=
1 !       constructs require a transfer of control to the
1 !       user program rather than an an error return.
1 !       A third handler, FOR$$IOSTAT_HND is for auxilliary I/O statements
1 !       which either undwind with R0 containing an IOSTAT value or
1 !       resignal.
1 !       An argument specifies the cleanup to be performed if UNWIND occurs.
1 !
1 ! ENVIRONMENT:  User mode, AST level or not or mixed.
1 !               Note: this module is both shared (with no entry vectors) and non-shared
1 !               if FORTRAN compatibility routines call.
1 !
1 ! AUTHOR: Thomas N. Hastings, CREATION DATE: 03-Jun-77
1 !
1 ! MODIFIED BY:
1 !
1 !       Thomas N. Hastings, 03-Jun-77: VERSION 01
1 !       Steven B. Lionel, VAX/VMS V2.0
```

```
1 ! [Previous edit history deleted.  SBL 30-Sep-1982]
1 ! 1-019 - Look at FAO_COUNT in signal list to see where USER_PC is.  SBL 10-NOV-1980
1 ! 1-020 - Reset RAB$L_UBF and RAB$W_USZ in CLEANUP_LUB.  JAW 08-Jun-1981
1 ! 1-021 - Change OTS$$ data structure references to FOR$$.  SBL 30-Sep-1982
1 ! 1-022 - Look at FAB$W_IFI instead of LUB$W_IFI in CLEANUP_LUB.  QAR #1229.
1 !             SBL 7-Mar-1984
1 !--
1
```

```
;    67          0066  1  !
;    68          0067  1  ! PROLOGUE FILE:
;    69          0068  1  !
;    70          0069  1
;    71          0070  1  REQUIRE 'RTLIN:FORPROLOG';                              ! FORTRAN definitions
;    72          0136  1
;    73          0137  1  !
;    74          0138  1  ! TABLE OF CONTENTS:
;    75          0139  1  !
;    76          0140  1
;    77          0141  1  FORWARD ROUTINE
;    78          0142  1      FOR$$ERR_OPECLO,                                    ! Error handler for OPEN/CLOSE
;    79          0143  1      FOR$$ERR_ENDHND,                                    ! ERR=/END= handler for I/O statements
;    80          0144  1      FOR$$IOSTAT_HND,                                    ! IOSTAT only handler
;    81          0145  1      FOR$$IO_IN_PROG,                                    ! I/O in progress handler
;    82          0146  1      CLEANUP_LUB : NOVALUE;                              ! Perform appropriate LUB cleanup if UNWIND.
;    83          0147  1                                                          ! signal list.
;    84          0148  1
;    85          0149  1  !
;    86          0150  1  ! EQUATED SYMBOLS:
;    87          0151  1  !
;    88          0152  1  !       NONE
;    89          0153  1  !
;    90          0154  1  ! OWN STORAGE:
;    91          0155  1  !
;    92          0156  1  !       NONE
;    93          0157  1  !
;    94          0158  1  ! EXTERNAL REFERENCES:
;    95          0159  1  !
;    96          0160  1  !+
;    97          0161  1  ! MAINTENANCE NOTE:  Since this module is called by FORTRAN compatibility
;    98          0162  1  ! routines which are un-shared and the entry points are not vectored,
;    99          0163  1  ! a separate copy of this module is linked with the user program when
;   100          0164  1  ! the user calls a FORTRAN compatibility routine.  In order to prevent
;   101          0165  1  ! data truncation errors from the linker, all external references are
;   102          0166  1  ! of addressing mode general (rather than word displacement) even for
;   103          0167  1  ! the same PSECT.
;   104          0168  1  !-
;   105          0169  1
;   106          0170  1  EXTERNAL ROUTINE
;   107          0171  1      FOR$$CB_GET : JSB_CB_GET NOVALUE,                   ! Get current LUB/ISB/RAB
;   108          0172  1                                                          ! Note: this non-shared routine is loaded if
;   109          0173  1                                                          ! compatibility routines call, so can't reference
;   110          0174  1                                                          ! FOR$$A_CUR_LUB directly.
;   111          0175  1      FOR$$CB_POP : JSB_CB_POP NOVALUE,                   ! Pop current LUB/ISB/RAB
;   112          0176  1                                                          ! as specified by CCB.
;   113          0177  1      FOR$$FP_MATCH : CALL_CCB NOVALUE,                   ! Match FP in ISB chain
;   114          0178  1      FOR$$FREE_VM,                                       ! Free virtual memory
;   115          0179  1      FOR$$CLOSE_FILE,                                    ! RMS Close a file
;   116          0180  1      FOR$$SIG_FATINT : NOVALUE,                          ! SIGNAL_STOP OTS$_FATINTERR
;   117          0181  1      FOR$$SIG_DATCOR : NOVALUE,                          ! SIGNAL_STOP OTS$_INTDATCOR
;   118          0182  1                                                          ! (FATAL INTERNAL ERROR IN RUN-TIME LIBRARY)
;   119          0183  1      LIB$SIG_TO_RET;                                     ! convert a SIGNAL to error return
;   120          0184  1                                                          ! to caller of establisher with R0 set to signal value.
;   121          0185  1
```

```
1 GLOBAL ROUTINE FOR$$ERR_OPECLO (                          ! Error condition handler for OPEN/CLOSE
1         SIG_ARGS_ADR,                                     ! Adr. of SIGNAL args
1         MCH_ARGS_ADR,                                     ! Adr. of mechanism args
1         ENB_ARGS_ADR)                                     ! Adr.  of ENABLE declared args
1     =                                                     ! Condition handlers always have values
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       FOR$$ERR_OPECLO is an error conditon handler established by
1 !       the OPEN and CLOSE statement procedures.  If the user specified
1 !       an ERR= keyword parameter, the handler unwinds the stack after
1 !       storing the signaled error condition in the saved image of R0.
1 !       Otherwise, FOR$$ERR_OPECLO just resignals by simply returning
1 !       SS$_RESIGNAL (to CHF).
1 !       If and when an UNWIND occurs, the ENABLE arg UNWIND_ACT_ADR
1 !       specifies whether the LUB/ISB/RAB is to be pop, returned, or no-opped.
1 !       It is not popped if it had not yet been pushed as indicated
1 !       by the ENABLE arg UNWIND_ACT_ADR.
1 !
1 !       If ERR= and IOSTAT were both specified, then the returned
1 !       value is the FORTRAN small integer error code.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       SIG-ARG-ADR
1 !       SIG_ARGS_ADR.rl.ra        Adr. of Signal arg list
1 !       MCH_ARGS_ADR.rl.ra        Adr. of mechanism arg list
1 !       ENB_ARGS_ADR.rl.ra        Adr. of ENABLE arg list which contains:
1 !           ENABLE_COUNT.rbu.v    No. of longword following in ENABLE arg list
1 !           UNWIND_ACT_ADR.rl.r   Adr. of longword containing UNWIND action code.
1 !                                 Any of FOR$K_UNWINDNOP, FOR$K_UNWINDPOP,
1 !                                 FOR$K_UNWINDRET.
1 !           [OPECLO_ADR.rlu.ra]   Optional adr. of cannonical array of OPEN or CLOSE keyword
1 !                                 parameters after the encoded user parameter
1 !                                 list has been scanned and expanded into it.
1 !                                 Symbolic offsets into ENB_ARGS_ADR[1,OPEN$K_name] are of the
1 !                                 form OPEN$K_name as defined in FOROPN REQUIRE file.
1 !                                 If ommitted, assume no ERR= (DEFINE FILE, REWIND, etc)
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       FOR$$A_CUR_LUB            Adr. of current LUB/ISB/RAB or 0
1 !                                 Note: obtained by calling FOR$$CB_GET
1 !                                       rather than directly.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       SIG_ARGS_ADR[SIG$_USER_PC]      Set to user call PC to RTL
1 !
1 ! COMPLETION CODES:
1 !
1 !       SS$_RESIGNAL if no ERR= was specified
1 !       SS$_NORMAL if ERR= was specified (ignored by CHF on UNWIND)
1 !
1 ! SIDE EFFECTS:
1 !
```

```
1 !      If the user has specified ERR=, the stack is unwound to the
1 !      caller of the establisher (i.e., the user program) with the save image
1 !      of R0 set to the error status.
1 !      If no ERR= was specified,the error conditon is resignaled.
1 !      If UNWIND call, the current LUB/ISB/RAB may be popped or returned.
1 !--
1
2      BEGIN
2
2      BUILTIN
2          CALLG,
2          AP;
2
2      LITERAL                                          ! Define ENABLE arglist offsets
2          ENABLE_COUNT = 0,                            ! Offset in ENB_ARGS_ADR of no. of enable args following
2          UNWIND_ACT_ADR = 1,                          ! Adr. of longword containing
2                                                       ! UNWIND action code.
2          OPECLO_ADR = 2;                              ! Adr. of OPEN/CLOSE cannonical array
2
2      MAP
2          SIG_ARGS_ADR : REF BLOCK [, BYTE],          ! SIGNAL args
2          MCH_ARGS_ADR : REF BLOCK [, BYTE],          ! mechanism args
2          ENB_ARGS_ADR : REF VECTOR [OPECLO_ADR, LONG];   !ENABLE args list array
2
2      LOCAL
2          EST_FP : REF BLOCK [, BYTE],                ! Establisher's FP
2          SIG_PC_LOC: REF VECTOR [, LONG],            ! Location of user PC in signal list
2          OPECLO_ARRAY : REF VECTOR [OPEN$K_KEY_MAX + 1, LONG];   ! OPEN/CLOSE cannonical array
2
2      !+
2      ! If this is unwind condition, perform cleanup. since
2      ! Perform LUB cleanup indicated by EBABLE arg UNWIND_ACT_ADR
2      ! (set by the establisher).
2      !-
2
3      IF .BLOCK [SIG_ARGS_ADR [CHF$L_SIG_NAME], STS$V_COND_ID;, BYTE] EQL (SS$_UNWIND^-3)
2      THEN
3          BEGIN
3          CLEANUP_LUB (..ENB_ARGS_ADR [UNWIND_ACT_ADR]);
3          RETURN SS$_NORMAL;
2          END;
2
2      OPECLO_ARRAY = .ENB_ARGS_ADR [OPECLO_ADR];
2
2      !+
2      ! If this is not a FOR$ error or if another RTL handler has seen this
2      ! error (noted by signal argument for user PC being non-zero) then
2      ! just resignal.
2      !-
2
2      IF .BLOCK [SIG_ARGS_ADR [CHF$L_SIG_NAME], STS$V_FAC_NO;, BYTE] NEQ FOR$K_FAC_NO
2      THEN
2          RETURN SS$_RESIGNAL;
2
2      SIG_PC_LOC = SIG_ARGS_ADR [CHF$L_SIG_ARG1] + (.SIG_ARGS_ADR [CHF$L_SIG_ARG1] * %UPVAL);
2      IF .SIG_PC_LOC [0] NEQ 0
2      THEN
```

```
                RETURN SS$_RESIGNAL;

        !+
        ! Check if user provided ERR= keyword or not.  If yes, convert signal to
        ! a return to the caller of the establisher with condition value in R0.
        ! If IOSTAT is present, act as if ERR= is also.
        ! If caller omitted OPECLO_ADR entry in ENB_ARGS_ADR, treat as if no ERR=.
        !-

        IF .ENB_ARGS_ADR [ENABLE_COUNT] GEQU OPECLO_ADR AND (.OPECLO_ARRAY [OPEN$K_ERR] OR .OPECLO_ARRAY [
                OPEN$K_IOSTAT]) NEQ 0
        THEN
            BEGIN

            !+
            ! If IOSTAT was specified, store the value.
            !-

            IF .OPECLO_ARRAY [OPEN$K_IOSTAT] NEQ 0
            THEN
                BEGIN

                LOCAL
                    IOSTAT;

                IOSTAT = .BLOCK [SIG_ARGS_ADR [CHF$L_SIG_NAME], STS$V_CODE;, BYTE];

                IF .OPECLO_ARRAY [OPEN$K_IOSTAT_L]
                THEN
                    .OPECLO_ARRAY [OPEN$K_IOSTAT] = .IOSTAT
                ELSE
                    BEGIN

                    LOCAL
                        IOSTAT_ADR : REF BLOCK [, BYTE];

                    IOSTAT_ADR = .OPECLO_ARRAY [OPEN$K_IOSTAT];
                    IOSTAT_ADR [0, 0, 16, 0] = .IOSTAT;
                    END;

                END;

            IF NOT CALLG (.AP, LIB$SIG_TO_RET) THEN FOR$$SIG_FATINT ()

            END
        ELSE

        !+
        ! No ERR=, so set user call PC saved in stack frame of establisher and RESIGNAL
        !-

            BEGIN
            EST_FP = .MCH_ARGS_ADR [CHF$L_MCH_FRAME];
            SIG_PC_LOC [0] = .EST_FP [SF$L_SAVE_PC];

            END;                                        ! End no ERR=
;   294        0357  2      !+
;   295        0358  2      ! Return resignal condition (ignored if SYS$UNWIND called).
;   296        0359  2      !-
;   297        0360  2
;   298        0361  2      RETURN SS$_RESIGNAL
;   299        0362  1      END;                                                  ! End of FOR$$ERR_OPECLO handler


                                                                                  .TITLE  FOR$$ERROR Internal FORTRAN error handling modu
;                                                                                                             le
                                                                                  .IDENT  \1-022\

                                                                                  .EXTRN  FOR$$CB_GET, FOR$$CB_POP
                                                                                  .EXTRN  FOR$$FP_MATCH, FOR$$FREE_VM
                                                                                  .EXTRN  FOR$$CLOSE_FILE
                                                                                  .EXTRN  FOR$$SIG_FATINT
                                                                                  .EXTRN  FOR$$SIG_DATCOR
                                                                                  .EXTRN  LIB$SIG_TO_RET

                                                                                  .PSECT  _FOR$CODE,NOWRT,  SHR,  PIC,2

                                                         000C 00000               .ENTRY  FOR$$ERR_OPECLO, Save R2,R3                     ; 0186
                                      52        04    AC   D0 00002               MOVL    SIG_ARGS_ADR, R2                                ; 0278
00000124   8F        04    A2         19              03   ED 00006               CMPZV   #3, #25, 4(R2), #292                            ;
                                      10              12   10 00010               BNEQ    1$                                              ;
                                      50        0C    AC   D0 00012               MOVL    ENB_ARGS_ADR, R0                                ; 0281
                                                04    B0   DD 00016               PUSHL   @4(R0)                                          ;
                            0000V     CF              01   FB 00019               CALLS   #1, CLEANUP_LUB                                 ;
                                      50              01   D0 0001E               MOVL    #1, R0                                          ; 0282
                                                           04 00021               RET                                                     ;
                                      53        0C    AC   D0 00022  1$:          MOVL    ENB_ARGS_ADR, R3                                ; 0285
                                      50        08    A3   D0 00026               MOVL    8(R3), OPECLO_ARRAY                             ;
           18        06    A2         0C              00   ED 0002A               CMPZV   #0, #12, 6(R2), #24                             ; 0293
                                      54              12      00030               BNEQ    5$                                              ;
                                      51        08    A2   D0 00032               MOVL    8(R2), R1                                       ; 0297
                                      51        08  A241   DE 00036               MOVAL   8(R2)[R1], SIG_PC_LOC                           ;
                                                      61   D5 0003B               TSTL    (SIG_PC_LOC)                                    ; 0298
                                                      47   12 0003D               BNEQ    5$                                              ;
                                      02              63   D1 0003F               CMPL    (R3), #2                                        ; 0309
                                                      36   1F 00042               BLSSU   4$                                              ;
                            53        0C    A0  58    A0   C9 00044               BISL3   88(OPECLO_ARRAY), 12(OPECLO_ARRAY), R3          ;
                                                      2E   13 0004A               BEQL    4$                                              ; 0310
                                      53        58    A0   D0 0004C               MOVL    88(OPECLO_ARRAY), R3                            ; 0318
                                                      15   13 00050               BEQL    3$                                              ;
           52        04    A2         0C              03   EF 00052               EXTZV   #3, #12, 4(R2), IOSTAT                          ; 0325
                                      05        64    A0   E9 00058               BLBC    100(OPECLO_ARRAY), 2$                           ; 0327
                                      63              52   D0 0005C               MOVL    IOSTAT, (R3)                                    ; 0329
                                                      06   11 0005F               BRB     3$                                              ;
                                      50              53   D0 00061  2$:          MOVL    R3, IOSTAT_ADR                                  ; 0336
                                      60              52   B0 00064               MOVW    IOSTAT, (IOSTAT_ADR)                            ; 0337
                        00000000G     00              6C   FA 00067  3$:          CALLG   (AP), LIB$SIG_TO_RET                            ; 0342
                                      15              50   E8 0006E               BLBS    R0, 5$                                          ;
                        00000000G     00              00   FB 00071               CALLS   #0, FOR$$SIG_FATINT                             ;
                                                      0C   11 00078               BRB     5$                                              ;
                                      50        08    AC   D0 0007A  4$:          MOVL    MCH_ARGS_ADR, R0                                ; 0352
                                      50        04    A0   D0 0007E               MOVL    4(R0), EST_FP                                   ;
                                      61        10    A0   D0 00082               MOVL    16(EST_FP), (SIG_PC_LOC)                        ; 0353
```

```
                                 50     0918   8F  3C 00086 5$:      MOVZWL  #2328, R0                              ; 0361
                                                   04 0008B          RET                                            ; 0362

; Routine Size:  140 bytes,    Routine Base:  _FOR$CODE + 0000


;  300          0363  1
```

```
GLOBAL ROUTINE FOR$$ERR_ENDHND (                         ! FORTRAN I/O statement ERR=/END= error condition handler
        SIG_ARGS_ADR,                                    ! Adr. of signal arg list
        MCH_ARGS_ADR,                                    ! Adr. of mechanism arg list
        ENB_ARGS_ADR)                                    ! Adr. of ENABLE arg list
    =                                                    ! Return status for a condition handler

!++
! FUNCTIONAL DESCRIPTION:
!
!       FOR$$ERR_ENDHND is an error condition handler established
!       by each I/O statement which has an  ERR= and END= error transfer
!       mechanism (as an option of the user program).
!
!       If the signaled condition is FOR$_ENDDURREA (24='END-OF_FILE DURING READ')
!       and an END= has been specified by the user in his I/O statement
!       (.END_EQL_ADR NEQ 0), the handler unwinds to the user specified address (by calling
!       SYS$UNWIND with depth equal to CHF$L_MCH_DEPTH + ..INCR_DEPTH_ADR + 1)
!       and new_PC equal to ..END_EQL_ADR.
!       Otherwise, if an ERR= had been specified by the user in his I/O statement
!       (ERR_EQUL NEQ 0), the handler unwinds to the user specified address
!       by calling SYS$UNWIND with depth equal to CHF$L_MCH_DEPTH + ..INCR_DEPTH_ADR + 1
!       and new_PC equal to ..ERR_EQL_ADR.
!
!       If neither of the above cases holds, the error is resignaled
!       so that a user handler or the OTS default handler will get invoked.
!       If UNWIND occurs, the appropriate cleanup takes place,
!       as indicated by the establisher in the ENABLE arg UNWIND_ACT_ADR.
!       If FOR$K_UNWINDPOP is indicated, the current LUB/ISB/RAB is popped.
!       If FOR$K_UNWINDRET is indicated, the LUB/ISB/RAB is returned and the
!       file closed.
!       Otherwise (FOR$K_UNWINDNOP) nothing is done.
!
! FORMAL PARAMETERS:
!
!       SIG_ARGS_ADR.ml.ra         Adr. of signal arg list
!       MCH_ARGS_ADR.ml.ra         Adr. of mechanism arg list
!       ENB_ARGS_ADR.ml.ra         Adr. of ENABLE arg list which contains:
!           UNWIND_ACT_ADR.rl.r Adr. of longword contining UNWIND action code.
!                                  Any of FOR$K_UNWINDNOP, FOR$K_UNWINDPOP,
!                                  FOR$K_UNWINDRET.
!           ERR_EQL_ADR.ra.r       Adr. of longword containing Adr. of the user address
!                                  to be transferred to or 0 on any error condition
!           END_EQL_ADR.ra.r       Adr of longword containing Adr. of the user address
!                                  to be transferred to or 0 on end-of-file
!           INCR_DEPTH_ADR.rl.r Adr. of longword containing Incremental no. of frames between the establishe
!                                  and the users program (usually 0 or 1).
! Note: All parameters to a condition handler must be addresses of values in BLISS if used in an ENABLE.
!
! IMPLICIT INPUTS:
!
!       FOR$$A_CUR_LUB             Adr. of current LUB/ISB/RAB or 0
!                                  Note: obtained by calling FOR$$CB_GET rather than directly.
!
! IMPLICIT OUTPUTS:
!
!       SIG_ARGS_ADR[SIG$_USER_PC]       Set to user call PC to RTL
!
```

```
1 ! COMPLETION CODES:
1 !
1 !       SS$_RESIGNAL if no ERR= or END= was specified by user, so that
1 !       a user handler or the default OTS handler will get a chance.
1 !       SS$_NORMAL if unwind called (although ignored if unwind called)
1 !
1 ! SIDE EFFECTS:
1 !
1 !       If END= and EOF OR ERR= was specified, the stack is unwound
1 !       to user and new_PC is set from ..END_EQL_ADR or .ERR_EQL_ADR.
1 !       If unwind, the current LUB/ISB/RAB may be popped or returned.
1 !--
1
2     BEGIN
2
2     LOCAL
2         EST_FP : REF BLOCK [, BYTE],               ! Establisher's FP
2         SIG_PC_LOC: REF VECTOR [, LONG];           ! Location of user PC in signal list
2
2     LITERAL                                        ! Declare offsets in ENABLE VECTOR arg list
2         UNWIND_ACT_ADR = 1,                        ! UNWIND action code
2         ERR_EQL_ADR = 2,                           ! ERR= adr or 0
2         END_EQL_ADR = 3,                           ! END= adr or 0
2         INCR_DEPTH_ADR = 4;                        ! incremental depth
2
2     MAP
2         SIG_ARGS_ADR : REF BLOCK [, BYTE],        ! SIGNAL arg list
2         MCH_ARGS_ADR : REF BLOCK [, BYTE],        ! mechanism arg list
2         ENB_ARGS_ADR : REF VECTOR [INCR_DEPTH_ADR + 1, LONG];    ! ENABLE arg list
2
2
2     !+
2     ! Check for unwinding since handler gets called when it does an unwind.
2     ! If unwind, perform cleanup indicated by ENABLE arg UNWIND_ACT_ADR.
2     ! Then return to the unwinder to keep unwinding (return value ignored).
2     !-
2
3     IF .BLOCK [SIG_ARGS_ADR [CHF$L_SIG_NAME], STS$V_COND_ID;, BYTE] EQL (SS$_UNWIND^-3)
2     THEN
3         BEGIN
3         CLEANUP_LUB (..ENB_ARGS_ADR [UNWIND_ACT_ADR]);
3         RETURN SS$_NORMAL;
2         END;
2
2     !+
2     ! If error is not a FOR$ error or if another RTL handler has seen
2     ! this error then resignal.
2     !-
2
2     IF .BLOCK [SIG_ARGS_ADR [CHF$L_SIG_NAME], STS$V_FAC_NO;, BYTE] NEQ FOR$K_FAC_NO
2     THEN
2         RETURN SS$_RESIGNAL;
2
2     SIG_PC_LOC = SIG_ARGS_ADR [CHF$L_SIG_ARG1] + (.SIG_ARGS_ADR [CHF$L_SIG_ARG1] * %UPVAL);
2     IF .SIG_PC_LOC [0] NEQ 0
2     THEN
2         RETURN SS$_RESIGNAL;
```

```
2
2       !+
2       ! Check for END= and ERR=.
2       ! If this is end-of-file (during read)
2       ! Unwind to the user with the new_pc being .END_ADR and with
2       ! R0 as an IOSTAT value of -1.
2       !-
2
2       IF ..ENB_ARGS_ADR [END_EQL_ADR] NEQA 0 AND .SIG_ARGS_ADR [CHF$L_SIG_NAME] EQL FOR$_ENDDURREA
2       THEN
3           BEGIN
3
3           LOCAL
3               T;
3
3           MCH_ARGS_ADR [CHF$L_MCH_SAVR0] = -1;
3           T = .MCH_ARGS_ADR [CHF$L_MCH_DEPTH] + ..ENB_ARGS_ADR [INCR_DEPTH_ADR] + 1;
3
4           IF $UNWIND (DEPADR = T, NEWPC = ..ENB_ARGS_ADR [END_EQL_ADR])
3           THEN
3               RETURN SS$_NORMAL
3           ELSE
3               FOR$$SIG_FATINT ()
3
2           END;
2
2       !+
2       ! If this is an error, and ERR= was specified by the user,
2       ! Unwind to the user with the new-pc being .ERR_ADR and
2       ! with R0 set to the proper IOSTAT value.
2       !-
2
2       IF ..ENB_ARGS_ADR [ERR_EQL_ADR] NEQA 0
2       THEN
3           BEGIN
3
3           LOCAL
3               T;
3
3           IF .SIG_ARGS_ADR [CHF$L_SIG_NAME] EQL FOR$_ENDDURREA
3           THEN
3               MCH_ARGS_ADR [CHF$L_MCH_SAVR0] = -1
3           ELSE
3               MCH_ARGS_ADR [CHF$L_MCH_SAVR0] = .BLOCK [SIG_ARGS_ADR [CHF$L_SIG_NAME], STS$V_CODE;, BYTE];
3
3           T = .MCH_ARGS_ADR [CHF$L_MCH_DEPTH] + ..ENB_ARGS_ADR [INCR_DEPTH_ADR] + 1;
3
4           IF $UNWIND (DEPADR = T, NEWPC = ..ENB_ARGS_ADR [ERR_EQL_ADR])
3           THEN
3               RETURN SS$_NORMAL
3           ELSE
3               FOR$$SIG_FATINT ()
3
2           END;
2
2       !+
2       ! If neither END= nor ERR= specified by user.
```

```
;  473     0535  2      ! Scan back from frame of establisher to frame of routine to called by user.
;  474     0536  2      ! Set user CALL PC to library in SIGNAL arg list.
;  475     0537  2      ! Just indicate to the condition handling facility to resignal the condition
;  476     0538  2      ! so that a user supplied handler or the OTS default handler will get a chance to handle.
;  477     0539  2      !-
;  478     0540  2
;  479     0541  2      EST_FP = .MCH_ARGS_ADR [CHF$L_MCH_FRAME];
;  480     0542  2
;  481     0543  2      DECR I FROM ..ENB_ARGS_ADR [INCR_DEPTH_ADR] TO 1 DO
;  482     0544  2          EST_FP = .EST_FP [SF$L_SAVE_FP];
;  483     0545  2
;  484     0546  2      SIG_PC_LOC [0] = .EST_FP [SF$L_SAVE_PC];
;  485     0547  2
;  486     0548  2      RETURN SS$_RESIGNAL
;  487     0549  1      END;                                              !End of FOR$$ERR_ENDHND


                                                                      .EXTRN  SYS$UNWIND

                                               007C 00000          .ENTRY  FOR$$ERR_ENDHND, Save R2,R3,R4,R5,R6              ; 0364
                              56 00000000G  00  9E 00002          MOVAB   FOR$$SIG_FATINT, R6                               ;
                              55 00000000G  00  9E 00009          MOVAB   SYS$UNWIND, R5                                    ;
                              5E            08  C2 00010          SUBL2   #8, SP                                            ;
                              52        04  AC  D0 00013          MOVL    SIG_ARGS_ADR, R2                                  ; 0458
                              53        04  A2  9E 00017          MOVAB   4(R2), R3                                         ;
00000124    8F            63                19  03  ED 0001B          CMPZV   #3, #25, (R3), #292                               ;
                                            0F  12 00024          BNEQ    1$                                                ;
                              50        0C  AC  D0 00026          MOVL    ENB_ARGS_ADR, R0                                  ; 0461
                                        04  B0  DD 0002A          PUSHL   @4(R0)                                            ;
                       0000V  CF            01  FB 0002D          CALLS   #1, CLEANUP_LUB                                   ;
                                          0080  31 00032          BRW     7$                                                ; 0462
          18        02    A3          0C    00  ED 00035 1$:      CMPZV   #0, #12, 2(R3), #24                               ; 0470
                                            0B  12 0003B          BNEQ    2$                                                ;
                              50        08  A2  D0 0003D          MOVL    8(R2), R0                                         ; 0474
                              54        08 A240 DE 00041          MOVAL   8(R2)[R0], SIG_PC_LOC                             ;
                                            64  D5 00046          TSTL    (SIG_PC_LOC)                                      ; 0475
                                            03  13 00048 2$:      BEQL    3$                                                ;
                                          0089  31 0004A          BRW     12$                                               ;
                              52        0C  AC  D0 0004D 3$:      MOVL    ENB_ARGS_ADR, R2                                  ; 0486
                                        0C  B2  D5 00051          TSTL    @12(R2)                                           ;
                                            2A  13 00054          BEQL    4$                                                ;
                    001880C4  8F            63  D1 00056          CMPL    (R3), #1605828                                    ;
                                            21  12 0005D          BNEQ    4$                                                ;
                              50        08  AC  D0 0005F          MOVL    MCH_ARGS_ADR, R0                                  ; 0493
                        0C    A0            01  CE 00063          MNEGL   #1, 12(R0)                                        ;
                  50    08    A0        10  B2  C1 00067          ADDL3   @16(R2), 8(R0), R0                                ; 0494
                              6E        01  A0  9E 0006D          MOVAB   1(R0), T                                          ;
                                        0C  B2  DD 00071          PUSHL   @12(R2)                                           ; 0496
                                        04  AE  9F 00074          PUSHAB  T                                                 ;
                              65            02  FB 00077          CALLS   #2, SYS$UNWIND                                    ;
                              38            50  E8 0007A          BLBS    R0, 7$                                            ;
                              66            00  FB 0007D          CALLS   #0, FOR$$SIG_FATINT                               ; 0500
                                        08  B2  D5 00080 4$:      TSTL    @8(R2)                                            ; 0510
                                            37  13 00083          BEQL    9$                                                ;
                              50        08  AC  D0 00085          MOVL    MCH_ARGS_ADR, R0                                  ; 0519
                    001880C4  8F            63  D1 00089          CMPL    (R3), #1605828                                    ; 0517
```

```
                                          06  12 00090            BNEQ    5$                                   ;
                      0C    A0            01  CE 00092            MNEGL   #1, 12(R0)                           ; 0519
                                          06  11 00096            BRB     6$                                   ;
     0C    A0          63      0C         03  EF 00098 5$:        EXTZV   #3, #12, (R3), 12(R0)                ; 0521
                       50  08  A0     10  B2  C1 0009E 6$:        ADDL3   @16(R2), 8(R0), R0                   ; 0523
                           04  AE     01  A0  9E 000A4            MOVAB   1(R0), T                             ;
                                      08  B2  DD 000A9            PUSHL   @8(R2)                               ; 0525
                                      08  AE  9F 000AC            PUSHAB  T                                    ;
                               65         02  FB 000AF            CALLS   #2, SYS$UNWIND                       ;
                               04         50  E9 000B2            BLBC    R0, 8$                               ;
                               50         01  D0 000B5 7$:        MOVL    #1, R0                               ; 0527
                                          04     000B8            RET                                          ;
                               66         00  FB 000B9 8$:        CALLS   #0, FOR$$SIG_FATINT                  ; 0529
                               50     08  AC  D0 000BC 9$:        MOVL    MCH_ARGS_ADR, R0                     ; 0541
                               50     04  A0  D0 000C0            MOVL    4(R0), EST_FP                        ;
                       51  10  B2         01  C1 000C4            ADDL3   #1, @16(R2), I                       ; 0543
                                          04  11 000C9            BRB     11$                                  ;
                               50     0C  A0  D0 000CB 10$:       MOVL    12(EST_FP), EST_FP                   ; 0544
                               F9         51  F5 000CF 11$:       SOBGTR  I, 10$                               ;
                               64     10  A0  D0 000D2            MOVL    16(EST_FP), (SIG_PC_LOC)             ; 0546
                               50   0918  8F  3C 000D6 12$:       MOVZWL  #2328, R0                            ; 0548
                                          04     000DB            RET                                          ; 0549

; Routine Size:  220 bytes,    Routine Base:  _FOR$CODE + 008C


;  488          0550  1
```

```
1 GLOBAL ROUTINE FOR$$IOSTAT_HND (                              ! FORTRAN I/O IOSTAT handler
1         SIG_ARGS_ADR,                                         ! Adr. of signal arg list
1         MCH_ARGS_ADR,                                         ! Adr. of mechanism arg list
1         ENB_ARGS_ADR)                                         ! Adr. of ENABLE arg list
1     =                                                         ! Return status for a condition handler
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       FOR$$IOSTAT_HND is an error condition handler established by each
1 !       auxilliary I/O statement which can have as optional arguments
1 !       ERR= and IOSTAT=.
1 !
1 !       If the enable argument ERR_EQL_ADR is non zero, FOR$$IOSTAT_HND
1 !       unwinds with the saved R0 set to the appropriate IOSTAT small
1 !       integer FORTRAN error number.  If ERR_EQL_ADR is zero, then it
1 !       is assumed that no ERR= is present and the error is resignalled.
1 !       Note that the unwind is not done to the ERR= address, rather the
1 !       compiled code makes a test of the returned value and branches
1 !       to the designated ERR= statement itself.
1 !
1 !       If UNWIND occurs, the appropriate cleanup takes place,
1 !       as indicated by the establisher in the ENABLE arg UNWIND_ACT_ADR.
1 !       If FOR$K_UNWINDPOP is indicated, the current LUB/ISB/RAB is popped.
1 !       If FOR$K_UNWINDRET is indicated, the LUB/ISB/RAB is returned and the
1 !       file closed.
1 !       Otherwise (FOR$K_UNWINDNOP) nothing is done.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       SIG_ARGS_ADR.ml.ra      Adr. of signal arg list
1 !       MCH_ARGS_ADR.ml.ra      Adr. of mechanism arg list
1 !       ENB_ARGS_ADR.ml.ra      Adr. of ENABLE arg list which contains:
1 !           UNWIND_ACT_ADR.rl.r Adr. of longword contining UNWIND action code.
1 !                               Any of FOR$K_UNWINDNOP, FOR$K_UNWINDPOP,
1 !                               FOR$K_UNWINDRET.
1 !           ERR_EQL_ADR.rl.v    0 if there is no ERR= on the statement
1 !                               1 if there is an ERR= present.
1 ! Note: All parameters to a condition handler must be addresses of values in BLISS if used in an ENABLE.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       FOR$$A_CUR_LUB          Adr. of current LUB/ISB/RAB or 0
1 !                               Note: obtained by calling FOR$$CB_GET rather than directly.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       MCH_ARGS_ADR [CHF$L_MCH_SAVR0]  Set to an IOSTAT value
1 !
1 ! COMPLETION CODES:
1 !
1 !       SS$_RESIGNAL if no ERR= or END= was specified by user, so that
1 !       a user handler or the default OTS handler will get a chance.
1 !       SS$_NORMAL if unwind called (although ignored if unwind called)
1 !
1 ! SIDE EFFECTS:
1 !
```

```
1 !       If ERR= was specified, the stack is unwound to the user.
1 !       If unwind, the current LUB/ISB/RAB may be popped or returned.
1 !--
1
2     BEGIN
2
2     LOCAL
2         EST_FP : REF BLOCK [, BYTE],                ! Establisher's FP
2         SIG_PC_LOC: REF VECTOR [, LONG];            ! Location of user PC in signal list
2
2     LITERAL                                         ! Declare offsets in ENABLE VECTOR arg list
2         UNWIND_ACT_ADR = 1,                         ! UNWIND action code
2         ERR_EQL_ADR = 2;                            ! ERR= present, 1 or 0
2
2     MAP
2         SIG_ARGS_ADR : REF BLOCK [, BYTE],          ! SIGNAL arg list
2         MCH_ARGS_ADR : REF BLOCK [, BYTE],          ! mechanism arg list
2         ENB_ARGS_ADR : REF VECTOR [ERR_EQL_ADR + 1, LONG];      ! ENABLE arg list
2
2     !+
2     ! Check for unwinding since handler gets called when it does an unwind.
2     ! If unwind, perform cleanup indicated by ENABLE arg UNWIND_ACT_ADR.
2     ! Then return to the unwinder to keep unwinding (return value ignored).
2     !-
2
3     IF .BLOCK [SIG_ARGS_ADR [CHF$L_SIG_NAME], STS$V_COND_ID;, BYTE] EQL (SS$_UNWIND^-3)
2     THEN
3         BEGIN
3         CLEANUP_LUB (..ENB_ARGS_ADR [UNWIND_ACT_ADR]);
3         RETURN SS$_NORMAL;
2         END;
2
2     !+
2     ! If this is not a FOR$ error or if another RTL handler has seen this
2     ! error (noted by signal argument for user PC being non-zero) then
2     ! just resignal.
2     !-
2
2     IF .BLOCK [SIG_ARGS_ADR [CHF$L_SIG_NAME], STS$V_FAC_NO;, BYTE] NEQ FOR$K_FAC_NO
2     THEN
2         RETURN SS$_RESIGNAL;
2     SIG_PC_LOC = SIG_ARGS_ADR [CHF$L_SIG_ARG1] + (.SIG_ARGS_ADR [CHF$L_SIG_ARG1] * %UPVAL);
2     IF .SIG_PC_LOC [0] NEQ 0
2     THEN
2         RETURN SS$_RESIGNAL;
2
2     !+
2     ! If this is an error, and ERR= was specified by the user,
2     ! Unwind to the user with saved R0 being the IOSTAT value.
2     !-
2
2     IF ..ENB_ARGS_ADR [ERR_EQL_ADR] NEQA 0
2     THEN
3         BEGIN
3         MCH_ARGS_ADR [CHF$L_MCH_SAVR0] = .BLOCK [SIG_ARGS_ADR [CHF$L_SIG_NAME], STS$V_CODE;, BYTE];
3
4         IF $UNWIND ()
```

```
;  604       0665  3           THEN
;  605       0666  3               RETURN SS$_NORMAL
;  606       0667  3           ELSE
;  607       0668  3               FOR$$SIG_FATINT ();
;  608       0669  3
;  609       0670  2           END;
;  610       0671  2
;  611       0672  2       !+
;  612       0673  2       ! If ERR= not specified by the user
;  613       0674  2       ! scan back from frame of establisher to frame of routine to called by user.
;  614       0675  2       ! Set user CALL PC to library in SIGNAL arg list.
;  615       0676  2       ! Just indicate to the condition handling facility to resignal the condition
;  616       0677  2       ! so that a user supplied handler or the OTS default handler will get a chance to handle.
;  617       0678  2       !-
;  618       0679  2
;  619       0680  2       EST_FP = .MCH_ARGS_ADR [CHF$L_MCH_FRAME];
;  620       0681  2       SIG_PC_LOC [0] = .EST_FP [SF$L_SAVE_PC];
;  621       0682  2       RETURN SS$_RESIGNAL
;  622       0683  1       END;                                                    !End of FOR$$IOSTAT_HND



                                                  000C 00000         .ENTRY  FOR$$IOSTAT_HND, Save R2,R3                 ; 0551
                             52        04      AC   D0 00002         MOVL    SIG_ARGS_ADR, R2                            ; 0633
00000124   8F      04   A2   19                03   ED 00006         CMPZV   #3, #25, 4(R2), #292                        ;
                                                0E   12 00010         BNEQ    1$                                          ;
                             50        0C      AC   D0 00012         MOVL    ENB_ARGS_ADR, R0                            ; 0636
                                       04      B0   DD 00016         PUSHL   @4(R0)                                      ;
                      0000V  CF                01   FB 00019         CALLS   #1, CLEANUP_LUB                             ;
                                                35   11 0001E         BRB     2$                                          ; 0637
           18      06   A2   0C                00   ED 00020 1$:     CMPZV   #0, #12, 6(R2), #24                         ; 0646
                                                44   12 00026         BNEQ    5$                                          ;
                             50        08      A2   D0 00028         MOVL    8(R2), R0                                   ; 0649
                             53        08    A240   DE 0002C         MOVAL   8(R2)[R0], SIG_PC_LOC                       ;
                                                63   D5 00031         TSTL    (SIG_PC_LOC)                                ; 0650
                                                37   12 00033         BNEQ    5$                                          ;
                             50        0C      AC   D0 00035         MOVL    ENB_ARGS_ADR, R0                            ; 0659
                                       08      B0   D5 00039         TSTL    @8(R0)                                      ;
                                                22   13 0003C         BEQL    4$                                          ;
                             50        08      AC   D0 0003E         MOVL    MCH_ARGS_ADR, R0                            ; 0662
     0C    A0      04   A2   0C                03   EF 00042         EXTZV   #3, #12, 4(R2), 12(R0)                      ;
                                                7E   7C 00049         CLRQ    -(SP)                                       ; 0664
                  00000000G  00                02   FB 0004B         CALLS   #2, SYS$UNWIND                              ;
                             04                50   E9 00052         BLBC    R0, 3$                                      ;
                             50                01   D0 00055 2$:     MOVL    #1, R0                                      ; 0666
                                                    04 00058         RET                                                 ;
                  00000000G  00                00   FB 00059 3$:     CALLS   #0, FOR$$SIG_FATINT                         ; 0668
                             50        08      AC   D0 00060 4$:     MOVL    MCH_ARGS_ADR, R0                            ; 0680
                             50        04      A0   D0 00064         MOVL    4(R0), EST_FP                               ;
                             63        10      A0   D0 00068         MOVL    16(EST_FP), (SIG_PC_LOC)                    ; 0681
                             50      0918      8F   3C 0006C 5$:     MOVZWL  #2328, R0                                   ; 0682
                                                    04 00071         RET                                                 ; 0683

; Routine Size:  114 bytes,    Routine Base:  _FOR$CODE + 0168
;  623        0684  1
```

```
1 GLOBAL ROUTINE FOR$$IO_IN_PROG (                        ! I/O in progress handler
1         SIG_ARGS_ADR,                                   ! Address of signal arg list
1         MCH_ARGS_ADR)                                   ! Address of mechanism arg list
1     =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       FOR$$IO_IN_PROG is a special handler that is designed to
1 !       allow the Run-Time Library to clean I/O that is in progress
1 !       when an error occurs during the processing of a multi-call
1 !       I/O statement.  For example, if evaluation of a variable
1 !       list item in a WRITE statement causes an error to be signalled,
1 !       there is no RTL handler in the stack frame to catch the error
1 !       and clean up in the case of an unwind
1 !
1 !       This handler is enabled at the user's stack frame level.  The
1 !       address of whatever user handler that was in the frame is stored
1 !       in the ISB.  When an error is signalled, this handler finds
1 !       the address of the user handler, if any, and calls it.  There
1 !       should be no normally detectable difference caused by FOR$$IO_IN_PROG
1 !       being on the frame.  On unwind, the current ISB is popped and the
1 !       user's handler is called again.  This way, we are protected against
1 !       all errors on all call levels.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       SIG_ARGS_ADR.ml.ra      Address of signal arguments list
1 !       MCH_ARGS_ADR.ml.ra      Address of mechanism arguments list
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       ISB/LUB/RAB database
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       ISB/LUB/RAB database
1 !
1 ! COMPLETION CODES:
1 !
1 !       Whatever is returned by the user handler.
1 !--
1
2     BEGIN
2
2     GLOBAL REGISTER
2         CCB = 11 : REF $FOR$CCB_DECL;
2
2     BUILTIN
2         CALLG,
2         AP;
2
2     LOCAL
2         USER_HANDLER,                                   ! Address of user's handler
2         EST_FP : REF BLOCK [, BYTE];                    ! Establisher's FP
2
2     MAP
```

```
;  682        0742  2           SIG_ARGS_ADR : REF BLOCK [, BYTE],       ! signal argument list
;  683        0743  2           MCH_ARGS_ADR : REF BLOCK [, BYTE];       ! mechanism argument list
;  684        0744  2
;  685        0745  2       !+
;  686        0746  2       ! Get establisher's FP
;  687        0747  2       !-
;  688        0748  2
;  689        0749  2       EST_FP = .MCH_ARGS_ADR [CHF$L_MCH_FRAME];
;  690        0750  2
;  691        0751  2       !+
;  692        0752  2       ! See if we are unwinding.
;  693        0753  2       !-
;  694        0754  2
;  695        0755  3       IF .BLOCK [SIG_ARGS_ADR [CHF$L_SIG_NAME], STS$V_COND_ID;, BYTE] EQL (SS$_UNWIND^-3)
;  696        0756  2       THEN
;  697        0757  3           BEGIN
;  698        0758  3           FOR$$CB_GET ();                          ! Get address of current LUB
;  699        0759  3
;  700        0760  3           IF .EST_FP NEQ .CCB [ISB$A_USER_FP] THEN FOR$$SIG_FATINT ();      ! Error
;  701        0761  3
;  702        0762  3           USER_HANDLER = .CCB [ISB$A_USR_HANDL];   ! Get user's handler address
;  703        0763  3           CLEANUP_LUB (FOR$K_UNWINDPOP);           ! Clean up LUB and restore user's handler
;  704        0764  3
;  705        0765  3           IF .USER_HANDLER NEQ 0 THEN RETURN CALLG (.AP, .USER_HANDLER);
;  706        0766  3
;  707        0767  3           RETURN SS$_NORMAL;
;  708        0768  2           END;
;  709        0769  2
;  710        0770  2       !+
;  711        0771  2       ! This is a signal.  Find the ISB that matched the establisher's
;  712        0772  2       ! FP.
;  713        0773  2       !-
;  714        0774  2
;  715        0775  2       FOR$$FP_MATCH (.EST_FP);
;  716        0776  2
;  717        0777  2       !+
;  718        0778  2       ! Call user's handler and return.
;  719        0779  2       !-
;  720        0780  2
;  721        0781  2       USER_HANDLER = .CCB [ISB$A_USR_HANDL];
;  722        0782  2
;  723        0783  2       IF .USER_HANDLER NEQ 0 THEN RETURN CALLG (.AP, .USER_HANDLER) ELSE RETURN SS$_RESIGNAL;
;  724        0784  2
;  725        0785  1       END;                                         ! End of FOR$$IO_IN_PROG



                                            080C 00000         .ENTRY  FOR$$IO_IN_PROG, Save R2,R3,R11                    ; 0685
                    50        08        AC    D0 00002         MOVL    MCH_ARGS_ADR, R0                                   ; 0749
                    53        04        A0    D0 00006         MOVL    4(R0), EST_FP                                      ;
                    50        04        AC    D0 0000A         MOVL    SIG_ARGS_ADR, R0                                   ; 0755
00000124    8F        04    A0          19    03  ED 0000E         CMPZV   #3, #25, 4(R0), #292                               ;
                                            28    12 00018         BNEQ    2$                                                 ;
                              00000000G   00    16 0001A         JSB     FOR$$CB_GET                                        ; 0758
                    FF4C    CB            53    D1 00020         CMPL    EST_FP, -180(CCB)                                  ; 0760
```

```
                                         07  13 00025          BEQL    1$                                    ;
          00000000G  00                  00  FB 00027          CALLS   #0, FOR$$SIG_FATINT                   ;
                     52        FF44      CB  D0 0002E 1$:      MOVL    -188(CCB), USER_HANDLER               ; 0762
                                         7E  D4 00033          CLRL    -(SP)                                 ; 0763
              0000V  CF                  01  FB 00035          CALLS   #1, CLEANUP_LUB                       ;
                                         52  D5 0003A          TSTL    USER_HANDLER                          ; 0765
                                         14  12 0003C          BNEQ    3$                                    ;
                     50                  01  D0 0003E          MOVL    #1, R0                                ; 0767
                                             04 00041          RET                                           ;
                                         53  DD 00042 2$:      PUSHL   EST_FP                                ; 0775
          00000000G  00                  01  FB 00044          CALLS   #1, FOR$$FP_MATCH                     ;
                     52        FF44      CB  D0 0004B          MOVL    -188(CCB), USER_HANDLER               ; 0781
                                         04  13 00050          BEQL    4$                                    ; 0783
                     62                  6C  FA 00052 3$:      CALLG   (AP), (USER_HANDLER)                  ;
                                             04 00055          RET                                           ;
                     50        0918      8F  3C 00056 4$:      MOVZWL  #2328, R0                             ;
                                             04 0005B          RET                                           ; 0785

; Routine Size:  92 bytes,    Routine Base:  _FOR$CODE + 01DA

;  726          0786  1
```

```
1 ROUTINE CLEANUP_LUB (ACTION) : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Perform the UNWIND action indicated by ACTION on the current LUB.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       ACTION.rlu.v            FOR$K_UNWINDNOP, FOR$K_UNWINDPOP, or FOR$K_UNWINDRET.
1 !
1 !--
1
2     BEGIN
2
2     GLOBAL REGISTER
2         CCB = 11 : REF $FOR$CCB_DECL;
2
2     BIND
2         FAB = CCB: REF $FOR$FAB_CCB_STRUCT;
2
2     CASE .ACTION FROM FOR$K_UNWINDPOP TO FOR$K_UNWINDRET OF
2         SET
2
2         !+
2         ! If the UNWIND action is to pop the LUB/ISB/RAB, call CB_POP to do
2         ! the work.
2         !-
2
2         [FOR$K_UNWINDPOP] :
3             BEGIN
3
3             LOCAL
3                 USER_FP;                          ! User's FP
3
3             FOR$$CB_GET ();                       ! CCB set to adr. of current /LUB/ISB/RAB
3             USER_FP = .CCB [ISB$A_USER_FP];       ! Get user's FP
3
3             IF .USER_FP NEQ 0 THEN .USER_FP = .CCB [ISB$A_USR_HANDL];   ! Restore user's handler
3
3             CCB [RAB$L_UBF] = .CCB [LUB$A_RBUF_ADR];
3             CCB [RAB$W_USZ] = .CCB [LUB$W_RBUF_SIZE];
3             FOR$$CB_POP ();
2             END;
2
2         !+
2         ! If the UNWIND action is NOP, do nothing.
2         !-
2
2         [FOR$K_UNWINDNOP] :
2         ;
2
2         !+
2         ! If the UNWIND action is RET, then try to $CLOSE the file associated
2         ! with this LUB/ISB/RAB.  Deallocate any dynamic storage associated
2         ! with this LUB.  Return the LUB/ISB/RAB to free storage.
2
```

```
;  785       0844 2          [FOR$K_UNWINDRET] :
;  786       0845 3              BEGIN
;  787       0846 3              FOR$$CB_GET ();                              ! Set CCB to adr. of current LUB/ISB/RAB
;  788       0847 3    !+
;  789       0848 3    ! See if file is RMS opened.
;  790       0849 3    !-
;  791       0850 3
;  792       0851 4              IF (.FAB [FAB$W_IFI] NEQ 0)
;  793       0852 3              THEN
;  794       0853 3    !+
;  795       0854 3    ! Do an RMS Close of the file, and arrange to deallocate its LUB/ISB/RAB
;  796       0855 3    ! when all I/O to it is finished.  Normally, we are doing the only I/O
;  797       0856 3    ! to it.
;  798       0857 3    !-
;  799       0858 3                  FOR$$CLOSE_FILE ()
;  800       0859 3              ELSE
;  801       0860 3    !+
;  802       0861 3    ! Even though the file is not open, we wish to deallocate the LUB, since
;  803       0862 3    ! this is the simplest way to reinitialize it if the user tries to use
;  804       0863 3    ! the logical unit number again, so tell OTS$$POP_CCB to deallocate it.
;  805       0864 3    !-
;  806       0865 3                  CCB [LUB$V_DEALLOC] = 1;
;  807       0866 3
;  808       0867 3    !+
;  809       0868 3    ! We are done with the logical unit.
;  810       0869 3    !-
;  811       0870 3              FOR$$CB_POP ();
;  812       0871 2              END;
;  813       0872 2          TES;
;  814       0873 2
;  815       0874 1      END;




                                          0804 00000 CLEANUP_LUB:
                                                            .WORD   Save R2,R11                              ; 0787
                         52 00000000G  00  9E 00002          MOVAB   FOR$$CB_GET, R2
            02           00         04  AC  CF 00009          CASEL   ACTION, #0, #2                           ; 0808
          0020         003A       0006        0000E 1$:       .WORD   2$-1$,-                                  ;
                                                                      7$-1$,-                                  ;
                                                                      4$-1$                                    ;
                                        62  16 00014 2$:      JSB     FOR$$CB_GET                              ; 0822
                         50       FF4C  CB  D0 00016          MOVL    -180(CCB), USER_FP                       ; 0823
                                        05  13 0001B          BEQL    3$                                       ; 0825
                         60       FF44  CB  D0 0001D          MOVL    -188(CCB), (USER_FP)                     ;
                   24    AB         EC  AB  D0 00022 3$:      MOVL    -20(CCB), 36(CCB)                        ; 0827
                   20    AB         D2  AB  B0 00027          MOVW    -46(CCB), 32(CCB)                        ; 0828
                                        14  11 0002C          BRB     6$                                       ; 0829
                                        62  16 0002E 4$:      JSB     FOR$$CB_GET                              ; 0846
                                  46    AB  B5 00030          TSTW    70(FAB)                                  ; 0851
                                        09  13 00033          BEQL    5$                                       ;
               00000000G  00            00  FB 00035          CALLS   #0, FOR$$CLOSE_FILE                      ; 0858
                                        04  11 0003C          BRB     6$                                       ;
                         FF    AB       10  88 0003E 5$:      BISB2   #16, -1(FAB)                             ; 0865
                             00000000G  00  16 00042 6$:      JSB     FOR$$CB_POP                              ; 0870
```

```
                                                                         H  6
FOR$$ERROR       Internal FORTRAN error handling module           16-Sep-1984 00:20:31    VAX-11 Bliss-32 V4.0-742              Page  23
1-022                                                             14-Sep-1984 12:31:54    [FORRTL.SRC]FORERROR.B32;1                  (7)

                                                           04 00048 7$:       RET                                                ; 0874

; Routine Size:  73 bytes,    Routine Base:  _FOR$CODE + 0236

;  816          0875  1
;  817          0876  1 END                                                  !End of module
;  818          0877  1
;  819          0878  0 ELUDOM




;                                        PSECT SUMMARY
;
;       Name                        Bytes                       Attributes
;
;   _FOR$CODE                            639  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)



;                                Library Statistics
;
;                                                  -------- Symbols --------          Pages         Processing
;       File                                       Total    Loaded   Percent          Mapped        Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1               9776        18         0             581           00:01.0
;  _$255$DUA28:[FORRTL.OBJ]FORLIB.L32;1             711       190        26              52           00:00.6
;  _$255$DUA28:[FORRTL.OBJ]RTLLIB.L32;1              36         0         0               8           00:00.1




;                                         COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:FORERROR/OBJ=OBJ$:FORERROR MSRC$:FORERROR/UPDATE=(ENH$:FORERROR)

; Size:          639 code + 0 data bytes
; Run Time:          00:16.9
; Elapsed Time:      00:45.4
; Lines/CPU Min:      3120
; Lexemes/CPU-Min: 15828
; Memory Used:  119 pages
; Compilation Complete
```

0180 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY